Termination of Nondeterministic Quantum Programs

Yangjia Li, Nengkun Yu, and Mingsheng Ying

TNLIST, Dept. of CS, Tsinghua University
QCIS, FEIT, University of Technology, Sydney
liyangjia@gmail.com

Abstract

We define a language-independent model of nondeterministic quantum programs in which a quantum program consists of a finite set of quantum processes. These processes are represented by quantum Markov chains over the common state space. An execution of a nondeterministic quantum program is modeled by a sequence of actions of individual processes. These actions are described by super-operators on the state Hilbert space. At each step of an execution, a process is chosen nondeterministically to perform the next action.

A characterization of reachable space and a characterization of diverging states of a nondeterministic quantum program are presented. We establish a zero-one law for termination probability of the states in the reachable space of a nondeterministic quantum program. A combination of these results leads to a necessary and sufficient condition for termination of nondeterministic quantum programs. Based on this condition, an algorithm is found for checking termination of nondeterministic quantum programs within a fixed finite-dimensional state space.

A striking difference between nondeterministic classical and quantum programs is shown by example: it is possible that each of several quantum programs simulates the same classical program which terminates with probability 1, but the nondeterministic program consisting of them terminates with probability 0 due to the interference carried in the execution of them.

1 Introduction

Quantum algorithms are usually expressed at the very low-level of quantum circuits. As pointed out by Abramsky [1], high-level, conceptual methods are needed for designing, programming and reasoning about quantum computational systems. Along this line, intensive research on quantum programming has been conducted in the last 15 years. Several quantum programming lan-

guages have been defined, including QCL by Ömer [17], qGCL by Sanders and Zuliani [18], a quantum extension of C++ by Betteli et al. [5], QPL by Selinger [19], and QML by Altenkirch and Grattage [3]. The operational or denotational semantics of these languages have been introduced. D'Hondt and Panangaden [8] proposed the notion of quantum weakest precondition, and then a predicate transformer semantics of quantum programs was presented in [23]. Also, several proof systems for verification of quantum programs have been developed [4, 6, 7, 9, 22], and some approaches to the implementation of quantum programming languages have been suggested [16, 21, 27, 25]. Furthermore, several quantum process algebras have been proposed: CQP by Gay and Nagarajan [12], QAlg by Jorrand and Lalire [14] and qCCS [10], to model quantum communication and concurrency. For a more systematic exposition, we refer to two excellent survey papers [11, 20].

Nondeterminism provides an important high-level feature in classical computation for specifying programs' behavior, without having to specify details of implementations. Zuliani [26] found a way for embedding nondeterminism into his quantum programming language qGCL, and then used qGCL equipped with a nondeterministic choice constrct to model and reason about Mitchison and Josza's counterfactual computation [15] and quantum systems in mixed states. In this paper, we consider a class of nondeterministic quantum programs defined in a language-independent way. A nondeterministic quantum program consists of a collection of quantum processes. These processes are described by quantum Markov chains over the common state space. This model of nondeterministic quantum programs is indeed a quantum generalization of Markov decision processes, which are widely used in the studies of probabilistic programs, see for example [13].

This paper focuses on the termination problem of nondeterministic quantum programs within a fixed finite-dimensional state space. The paper is organized as follows. In Sec. 2, we briefly review the basic notions from quantum theory required in this paper, with an emphasis on fixing notations. In Sec. 3, a model of nondeterministic quantum programs is defined in terms of quantum Markov chains. In this model, an execution of a nondeterministic quantum program is a sequence of actions of individual processes, and following Selinger [19] these actions are depicted by super-operators on the state Hilbert space. At each step of an execution, a process is chosen nondeterministically to perform the next action. We define the termination probability of a nondeterministic quantum program starting in a state according to an execution schedule. Then the termination of a nondeterministic quantum program is defined to be the infimum of its termination probabilities over all possible schedules. At the end of this section, we consider an example of nondeterministic quantum program consisting of two quantum walks on a graph [2]. This example is interesting because it indicates a striking difference between nondeterministic classical and quantum programs: it is possible that each of several quantum programs simulates the same classical program which terminates with probability 1, but the nondeterministic program consisting of them terminates with probability 0 due to the interference carried in the execution of them. In Sec. 4, we examine the reach-

able space of a nondeterministic quantum program. By taking the arithmetic average of the super-operators performed by individual processes, we are able to define a deterministic quantum program whose reachable space is equal to the reachable space of the original nondeterministic program. Furthermore, the reachable space of the average deterministic program can be obtained by recursively constructing a finite increasing sequence of subspaces of the state Hilbert space. The notions of terminating and diverging states of a nondeterministic quantum program are introduced in Sec. 5. The structures of the sets of terminating and diverging states are clarified. In particular, it is shown that the space of diverging pure states can also be recursively constructed in a finite number of steps. In Sec. 6, the Hart-Sharir-Pnueli zero-one law for probabilistic concurrent programs [13] is generalized to the case of nondeterministic quantum programs. This quantum zero-one law enables us to discover an algorithmically checkable termination condition for nondeterministic quantum programs in terms of reachable space and diverging pure states. A classical (not quantum) algorithm for termination checking of nondeterministic quantum programs is then presented in Sec. 7. A brief conclusion is drawn in Sec. 8.

2 Preliminaries and Notations

We assume that the reader is familiar with basic quantum theory, and the main aim of this section is to fix notations.

2.1 Quantum states

In quantum mechanics, the state space of a physical system is described by a complex Hilbert space $\mathcal{H}$. In this paper, we only consider finite-dimensional Hilbert spaces. We write $\dim \mathcal{H}$ for the dimension of space $\mathcal{H}$. A pure state of a system is represented by a unit vector in the state space of the system, and a mixed state by a (partial) density operator, that is, a positive semi-definite matrix ρ with its trace $\mathrm{tr}(\rho) \leq 1$. We write $\mathcal{D}(\mathcal{H})$ for the set of (partial) density operators on $\mathcal{H}$. For convenience, we simply write ψ for the density operator corresponding to pure state $|\psi\rangle$, that is, $\psi = |\psi\rangle\langle\psi|$. For any two density operators ρ and σ, their distance is defined to be $\| \rho - \sigma \|_*$, where

$$\| M \|_* = \mathrm{tr}(\sqrt{MM^\dagger})$$

is the trace norm of M for all operators M. Let

$$\rho = \sum_i \lambda_i |\psi_i\rangle\langle\psi_i|$$

be the spectral decomposition of density operator ρ. The subspace $\mathrm{span}\{|\psi_i\rangle\}$ is called the support of ρ, written $\mathrm{supp}\rho$. Recall that for a family $\{X_i\}$ of subspaces of $\mathcal{H}$, the join of $\{X_i\}$ is defined by

$$\bigvee_i X_i = \mathrm{span}(\bigcup_i X_i),$$

and we write $X \vee Y$ for the join of two subspaces X and Y. Then it is easy to verify that for any two states $\rho, \sigma \in \mathcal{D}(\mathcal{H})$,

$$\text{supp}(\rho + \sigma) = \text{supp}(\rho) \vee \text{supp}(\sigma).$$

2.2 Quantum operations

Super-operators formalize physical transformations between quantum states. A super-operator $\mathcal{E}$ on a Hilbert space $\mathcal{H}$ is a linear map from linear operators on $\mathcal{H}$ to themselves satisfying the following two conditions:

1. Completely positive: for any a extra state space $\mathcal{H}'$ and any positive semi-definite operator $P \in \mathcal{D}(\mathcal{H}' \otimes \mathcal{H})$, $(\mathcal{I} \otimes \mathcal{E})(P)$ is always positive semi-definite, where $\mathcal{I} : \mathcal{B}(\mathcal{H}') \to \mathcal{B}(\mathcal{H}')$ is the identity super-operator;

2. Trace-preserving: for any $\rho \in \mathcal{D}(\mathcal{H})$, $\text{tr}(\mathcal{E}(\rho)) = \text{tr}(\rho)$.

The Kraus representation theorem asserts that a linear map $\mathcal{E}$ is a super-operator iff there are linear operators E_i such that

$$\mathcal{E}(\rho) = \sum_i E_i \rho E_i^\dagger$$

for all $\rho \in \mathcal{D}(\mathcal{H})$, and

$$\sum_i E_i^\dagger E_i = Id_{\mathcal{H}},$$

where $Id_{\mathcal{H}}$ is the identity operator on $\mathcal{H}$.

For any subspace X of $\mathcal{H}$, we define the image of X under $\mathcal{E}$ as

$$\mathcal{E}(X) = \bigvee_{|\psi\rangle \in X} \text{supp}\mathcal{E}(\psi).$$

In other words, if we write P_X for the projection operator of X, then $\mathcal{E}(X) = \text{supp}\mathcal{E}(P_X)$. Using the Kraus representation, it is easy to verify that $\mathcal{E}(\text{supp}(\rho)) = \text{supp}\mathcal{E}(\rho)$. We can also define the pre-image of X under $\mathcal{E}$ by

$$\mathcal{E}^{-1}(X) = \{|\psi\rangle \in \mathcal{H} | \text{supp}\mathcal{E}(\psi) \subseteq X\}.$$

It is actually the maximal subspace Y satisfying that $\mathcal{E}(Y) \subseteq X$. We write $X^\perp$ for the orthogonal complement of a subspace X, then it is also easy to verify that

$$\mathcal{E}^{-1}(X) = (\mathcal{E}^*(X^\perp))^\perp,$$

where the super-operator

$$\mathcal{E}^*(\cdot) = \sum_i E_i^\dagger \cdot E_i$$

is the Schrödinger-Heisenberg dual of

$$\mathcal{E}(\cdot) = \sum_i E_i \cdot E_i^\dagger.$$

2.3 Quantum measurements

To acquire information about a quantum system, a measurement must be performed on it. A quantum measurement on a system with state space $\mathcal{H}$ is described by a collection $\{M_i\}$ of linear operators on $\mathcal{H}$ satisfying

$$\sum_i M_i^\dagger M_i = Id_{\mathcal{H}},$$

where indices i stand for the outcomes that may occur in the experiment. If the system is in state $\rho \in \mathcal{D}(\mathcal{H})$ immediately before the measurement, then the probability that result i occurs is $\mathrm{tr}(M_i \rho M_i^\dagger)$, and the state of the system after the measurement is $M_i \rho M_i^\dagger$.

3 A Model of Nondeterministic Quantum Programs

3.1 Basic Definitions

Definition 1 *Let $\mathcal{H}$ be a finite-dimensional Hilbert space which will be used as the state space of programs. A nondeterministic quantum program is a pair*

$$\mathcal{P} = (\{\mathcal{E}_i : i = 1, \cdots, m\}, \{M_0, M_1\}),$$

where:

1. *$\mathcal{E}_i$ is a super-operator on $\mathcal{H}$ for each $i = 1, \cdots, m$;*
2. *$\{M_0, M_1\}$ is a measurement on $\mathcal{H}$.*

There are m processes in the program $\mathcal{P}$. The one-step running of process i is modeled by super-operator $\mathcal{E}_i$ for each $1 \leq i \leq m$. We will call $\mathcal{P}$ a deterministic quantum program when $m = 1$, that is, there is only one process in $\mathcal{P}$.

We now see how a nondeterministic quantum program be executed. We first consider a single computation step of program $\mathcal{P}$. It is achieved as follows:

- Before each step, the measurement $\{M_0, M_1\}$ is performed on the current state ρ to determine whether the program terminates or not. If the outcome is 0, the program terminates; Otherwise the program goes to complete a step then.
- In each step, an element i is nondeterministically chosen from the index set $\{1, 2, \cdots, m\}$ firstly, and then the operation $\mathcal{E}_i$ is performed on the current program state. Thus, the state becomes $\mathcal{E}_i(M_1 \rho M_1^\dagger)$ after the measurement and the operation $\mathcal{E}_i$.

A computation of a nondeterministic quantum program is a finite or infinite sequence of computation steps in which the same measurement is performed to

determine termination of the program in all steps, but the super-operators performed in different steps are usually different and they are nondeterministically scheduled. Formally, the set of schedules of program $\mathcal{P}$ is defined to be

$$\begin{aligned} S &= \{1, 2, \cdots, m\}^\infty \\ &= \{s_1 s_2 \cdots s_k \cdots : s_k \in \{1, 2, \cdots, m\} \text{ for all } k \geq 0\}. \end{aligned}$$

We also define the set of schedule fragments of $\mathcal{P}$ to be

$$S_{fin} = \{1, 2, \cdots, m\}^* = \bigcup_{n=0}^{\infty} \{1, 2, \cdots, m\}^n.$$

For convenience, we use ϵ to represent empty string. For any $f = s_1 \cdots s_m \in S_{fin}$, we write $|f|$ for the length of f, that is, $|f| = m$. For each $n \leq |f|$, $f(\leq n)$ stands for the head $s_1 \cdots s_n$ of f. We also write the head

$$s(\leq n) = s_1 s_2 \cdots s_n \in S_{fin}$$

and the tail

$$s(> n) = s_{n+1} s_{n+2} \cdots \in S$$

of $s = s_1 s_2 \cdots \in S$ for each $n \geq 0$. For any $s = s_1 s_2 \cdots \in S$ and $f = s'_1 \cdots s'_m \in S_{fin}$, we write fs for the concatenation of f and s, that is, the schedule

$$fs = s'_1 \cdots s'_m s_1 s_2 \cdots .$$

For simplicity of presentation, we introduce the notation $\mathcal{T}_i$ which stands for the super-operator defined by

$$\mathcal{T}_i(\rho) = \mathcal{E}_i(M_1 \rho M_1^\dagger)$$

for every $\rho \in \mathcal{D}(\mathcal{H})$ and $1 \leq i \leq m$. Furthermore, for any $f = s_1 s_2 \cdots s_n \in S_{fin}$, we write:

$$\mathcal{T}_f = \mathcal{T}_{s_n} \circ \cdots \circ \mathcal{T}_{s_2} \circ \mathcal{T}_{s_1},$$

in particular, $\mathcal{T}_\epsilon(\rho) = \rho$ for all ρ. Let $\rho \in \mathcal{D}(\mathcal{H})$. If the input is state ρ, and program $\mathcal{P}$ is executed according to a schedule $s = s_1 s_2 \cdots \in S$, the program state after n steps is $\mathcal{T}_{s(\leq n)}(\rho)$.

3.2 Termination Probability

Suppose that the input state to program $\mathcal{P}$ is ρ. For any schedule fragment $f \in S_{fin}$, we define the probability that the program terminates within f as follows:

$$t_f(\rho) = \sum_{n=0}^{|f|} \mathrm{tr}(M_0 \mathcal{T}_{f(\leq n)}(\rho) M_0^\dagger).$$

If the program is executed according to a schedule $s = s_1 s_2 \cdots$, it is easy to see that the probability of the program terminating in no more than n steps is

$t_{s(\leq n)}(\rho)$. Furthermore, the probability that the program terminates in a finite number of steps is

$$t_s(\rho) = \lim_{n\to\infty} t_{s(\leq n)}(\rho) = \sum_{n=0}^{\infty} \mathrm{tr}(M_0 \mathcal{T}_{s(\leq n)}(\rho) M_0^\dagger).$$

It is obvious that $tr(\rho) \geq t_s(\rho)$, and $tr(\rho) - t_s(\rho)$ is the divergence probability of the program starting in state ρ and executed according to schedule s. We can divide the termination probability $t_s(\rho)$ into two parts:

- The first part is the probability of terminating in less than n steps, that is

$$\begin{aligned} t_{s(\leq n-1)}(\rho) &= \sum_{k=0}^{n-1} \mathrm{tr}(M_0 \mathcal{T}_{s(\leq k)}(\rho) M_0^\dagger) \\ &= \mathrm{tr}(\rho) - \mathrm{tr}(M_1 \mathcal{T}_{s(\leq n-1)}(\rho) M_1^\dagger) \\ &= \mathrm{tr}(\rho) - \mathrm{tr}(\mathcal{T}_{s(\leq n)}(\rho)) \end{aligned} \tag{1}$$

  because all $\mathcal{E}_i$ $(i = 1, ..., m)$ are trace-preserving.

- The second part is the probability of terminating in at least n steps, that is

$$\begin{aligned} &\sum_{k=n}^{\infty} \mathrm{tr}(M_0 \mathcal{T}_{s(\leq k)}(\rho) M_0^\dagger) \\ &= \sum_{k=0}^{\infty} \mathrm{tr}(M_1 (\mathcal{T}_{s_{n+1} s_{n+2} \cdots s_{n+k}} \circ \mathcal{T}_{s(\leq n)})(\rho) M_1^\dagger) \\ &= t_{s(>n)}(\mathcal{T}_{s(\leq n)}(\rho)). \end{aligned} \tag{2}$$

Combining the above two equations, we get that

$$\mathrm{tr}(\rho) - t_s(\rho) = \mathrm{tr}(\mathcal{T}_{s(\leq n)}(\rho)) - t_{s(>n)}(\mathcal{T}_{s(\leq n)}(\rho)). \tag{3}$$

This indicates that the divergence probability of a program is an invariant through an execution path of the program.

In general, an execution along with any schedule $s \in S$ is possible for a nondeterministic program. So, we need to consider all possible execution paths of the program together.

Definition 2 *The termination probability of program $\mathcal{P}$ starting in state ρ is*

$$t(\rho) = \inf\{t_s(\rho) | s \in S\}.$$

3.3 An Example: Quantum Walks

We consider quantum walks on a graph. Let $C_4 = (V, E)$ be a circle with four vertices, where $V = \{0, 1, 2, 3\}$ is the set of vertices, and $E = \{(0, 1), (1, 2), (2, 3), (3, 0)\}$ is the set of edges. We first define a quantum walk on C_4 as follows:

- The state Hilbert space is $\mathbf{C}^V$, and it has $\{|i\rangle | i \in V\}$ as its computational basis;
- The initial state is $|0\rangle$. This means that the walk start at the vertex 0;
- A single step of the walk is defined by the unitary operator:

$$W_1 = \frac{1}{\sqrt{3}} \begin{pmatrix} 1 & 1 & 0 & -1 \\ 1 & -1 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 1 & 0 & -1 & 1 \end{pmatrix}.$$

  It means that at any vertex, the probability of walking to the left and walking to the right are both $1/3$, and there is also a probability $1/3$ of not walking.

- The termination measurement $\{P_0, P_1\}$ is defined by

$$P_0 = |2\rangle\langle 2|, \ P_1 = Id_4 - |2\rangle\langle 2|,$$

  which means that there is an absorbing boundary at vertex 2. Here, Id_4 is the 4×4 unit matrix.

This quantum walk can be seen as a deterministic quantum program $(W_1, \{P_0, P_1\})$ starting in state $|0\rangle$. It is easy to verify that this program terminates with probability 1, that is, $t(|0\rangle\langle 0|) = 1$. If the unitary operator W_1 in the above quantum walk is replaced by:

$$W_2 = \frac{1}{\sqrt{3}} \begin{pmatrix} 1 & 1 & 0 & 1 \\ -1 & 1 & -1 & 0 \\ 0 & 1 & 1 & -1 \\ 1 & 0 & -1 & -1 \end{pmatrix},$$

then we get a new quantum walk, which can also be seen as a deterministic quantum program $(W_2, \{P_0, P_1\})$ starting in state $|0\rangle$. This new quantum walk is terminating too. However, if we combine these two walks to form a nondeterministic quantum program $(\{W_1, W_2\}, \{P_0, P_1\})$, then it is not terminating when starting in state $|0\rangle$. In fact, since $W_2 W_1 |0\rangle = |0\rangle$, it holds that

$$t(|0\rangle\langle 0|) \leq t_s(|0\rangle\langle 0|) = 0$$

for the infinite execution path $s = (12)^\infty = 121212\cdots$.

4 Reachable Space

From now on, we consider a fixed nondeterministic quantum program

$$\mathcal{P} = (\{\mathcal{E}_i : i = 1, \cdots, m\}, \{M_0, M_1\}).$$

Definition 3 *1. The set of reachable states of program $\mathcal{P}$ starting in state ρ is*

$$R(\rho) = \{\mathcal{T}_f(\rho) | f \in S_{fin}\}.$$

2. The reachable space of program $\mathcal{P}$ starting in state ρ is the subspace of $\mathcal{H}$ spanned by $R(\rho)$, that is,

$$\mathcal{H}_{R(\rho)} = \bigvee \{\text{supp}\sigma | \sigma \in R(\rho)\}.$$

We imagine that during the running of $\mathcal{P}$, if each nondeterministic choice of $i \in \{1, 2, \cdots, m\}$ is made according to the uniform probability distribution, then then $\mathcal{P}$ actually implements a deterministic quantum program, which can be described as:

Definition 4 *The average of $\mathcal{P}$ is the deterministic quantum program*

$$\overline{\mathcal{P}} = (\mathcal{E}, \{M_0, M_1\}),$$

where $\{M_0, M_1\}$ is the same as in $\mathcal{P}$, and $\mathcal{E}$ is the arithmetic average of $\mathcal{E}_1, \mathcal{E}_2, \cdots, \mathcal{E}_m$, that is,

$$\mathcal{E}(\rho) = \frac{1}{m} \sum_{i=1}^{m} \mathcal{E}_i(\rho)$$

for every $\rho \in \mathcal{D}(\mathcal{H})$.

We define:

$$\mathcal{T}(\rho) = \mathcal{E}(M_1 \rho M_1^\dagger)$$

for every $\rho \in \mathcal{D}(\mathcal{H})$. Then the reachable set and reachable space of $\overline{\mathcal{P}}$ starting in state ρ are, respectively:

$$\overline{R}(\rho) = \{\mathcal{T}^n(\rho) | n = 0, 1, 2, ...\},$$
$$\mathcal{H}_{\overline{R}(\rho)} = \bigvee \{\text{supp}(\sigma) | \sigma \in \overline{R}(\rho)\}.$$

The following lemma shows that $\mathcal{P}$ and its average $\overline{\mathcal{P}}$ have the same reachable space.

Lemma 1 *For any $\rho \in \mathcal{D}(\mathcal{H})$, $\mathcal{H}_{R(\rho)} = \mathcal{H}_{\overline{R}(\rho)}$.*

Proof: For each $n \geq 0$, we have:

$$\begin{aligned}\mathcal{T}^n(\rho) &= (\frac{1}{m}\sum_{i=1}^{m}\mathcal{T}_i)^n(\rho)\\ &= \frac{1}{m^n}\sum_{s_1,s_2,\cdots,s_n\in S}(\mathcal{T}_{s_n}\circ\cdots\circ\mathcal{T}_{s_2}\circ\mathcal{T}_{s_1})(\rho)\\ &= \frac{1}{m^n}\sum_{f\in S_{fin},|f|=n}\mathcal{T}_f(\rho).\end{aligned}$$

Then the proof is completed by observing that

$$\begin{aligned}\mathcal{H}_{R(\rho)} &= \bigvee_{f\in S_{fin}}\mathrm{supp}(\mathcal{T}_f(\rho))\\ &= \bigvee_{n=0}^{\infty}\bigvee_{f\in S_{fin},|f|=n}\mathrm{supp}(\mathcal{T}_f(\rho))\\ &= \bigvee_{n=0}^{\infty}\mathrm{supp}(\mathcal{T}^n(\rho)) = \mathcal{H}_{\overline{R}(\rho)}.\ \square\end{aligned}$$

Now we only need to examine the reachable space $\mathcal{H}_{\overline{R}(\rho)}$. For every $n \geq 0$, we define $\mathcal{H}_{\overline{R}_n(\rho)}$ to be the reachable space of program $\overline{\mathcal{P}}$ within n steps when starting in state ρ, that is,

$$\mathcal{H}_{\overline{R}_n(\rho)} = \bigvee_{k=0}^{n}\mathrm{supp}(\mathcal{T}^k(\rho)).$$

Then it is clear that

$$\mathcal{H}_{\overline{R}(\rho)} = \bigvee_{n=0}^{\infty}\mathcal{H}_{\overline{R}_n(\rho)}.$$

On the other hand, all elements of the increasing chain

$$\mathcal{H}_{\overline{R}_1(\rho)} \subseteq \mathcal{H}_{\overline{R}_2(\rho)} \subseteq \cdots$$

are subspace of finite-dimensional space $\mathcal{H}$. There must be some $n \geq 0$ such that $\mathcal{H}_{\overline{R}_n(\rho)} = \mathcal{H}_{\overline{R}_k(\rho)}$ for all $k \geq n$, and thus $\mathcal{H}_{\overline{R}(\rho)} = \mathcal{H}_{\overline{R}_n(\rho)}$. Furthermore, we have a recursive characterization of reachable spaces $\mathcal{H}_{\overline{R}_n(\rho)}$.

Lemma 2 *For all $n \geq 0$, we have:*

$$\mathcal{H}_{\overline{R}_{n+1}(\rho)} = \mathcal{H}_{\overline{R}_n(\rho)} \vee \mathcal{T}(\mathcal{H}_{\overline{R}_n(\rho)}).$$

Proof: It holds that

$$\begin{aligned}\mathrm{supp}\mathcal{T}^{n+1}(\rho) &= \mathrm{supp}(\mathcal{T}(\mathcal{T}^n(\rho_n)))\\ &= \mathcal{T}(\mathrm{supp}(\mathcal{T}^n(\rho))) \subseteq \mathcal{T}(\mathcal{H}_{\mathcal{R}_n(\rho)}).\end{aligned}$$

So, we have:

$$\mathcal{H}_{\overline{R}_{n+1}(\rho)} \subseteq \mathcal{H}_{\mathcal{R}_n(\rho)} \vee \mathcal{T}(\mathcal{H}_{\mathcal{R}_n(\rho)}).$$

Conversely, it holds that

$$\begin{aligned}
&\mathcal{T}(\mathcal{H}_{\mathcal{R}_n(\rho)}) = \mathcal{T}(\mathrm{supp}(\sum_{k=0}^{n} \mathcal{T}^k(\rho))) \\
&= \mathrm{supp}(\mathcal{T}(\sum_{k=0}^{n} \mathcal{T}^k(\rho))) = \mathrm{supp}(\sum_{k=0}^{n} \mathcal{T}^{k+1}(\rho)) \\
&= \bigvee_{k=1}^{n+1} \mathrm{supp}(\mathcal{T}^k(\rho)) \subseteq \mathcal{H}_{\mathcal{R}_{n+1}(\rho)}.
\end{aligned}$$

Thus, we have:

$$\mathcal{H}_{\overline{R}_n(\rho)} \vee \mathcal{T}(\mathcal{H}_{\overline{R}_n}(\rho)) \subseteq \mathcal{H}_{\overline{R}_{n+1}(\rho)}. \square$$

Now, we are able to prove the main result in this section.

Theorem 1 *If n is the smallest integer n satisfying $\mathcal{H}_{\overline{R}_n(\rho)} = \mathcal{H}_{\overline{R}_{n+1}(\rho)}$, then $\mathcal{H}_{R(\rho)} = \mathcal{H}_{\overline{R}_n(\rho)}$.*

Proof: By Lemma 1, it suffices to show that

$$\mathcal{H}_{\overline{R}_n(\rho)} = \mathcal{H}_{\overline{R}_{n+1}(\rho)} \text{ implies } \mathcal{H}_{\overline{R}_{n+1}(\rho)} = \mathcal{H}_{\overline{R}_{n+2}(\rho)},$$

which in turn implies $\mathcal{H}_{\overline{R}_{n+k}(\rho)} = \mathcal{H}_{\overline{R}_{n+k+1}(\rho)}$ for all $k \geq 2$. In fact, it follows from Lemma 2 that

$$\begin{aligned}
\mathrm{supp}(\mathcal{T}^{n+2}(\rho)) &= \mathrm{supp}(\mathcal{T}(\mathcal{T}^{n+1}(\rho))) \\
&= \mathcal{T}(\mathrm{supp}(\mathcal{T}^{n+1}(\rho))) \subseteq \mathcal{T}(\mathcal{H}_{\overline{R}_{n+1}(\rho)}) \\
&= \mathcal{T}(\mathcal{H}_{\overline{R}_n}(\rho)) \subseteq \mathcal{H}_{\overline{R}_{n+1}(\rho)}.
\end{aligned}$$

Thus, we have $\mathcal{H}_{\overline{R}_{n+2}(\rho)} = \mathcal{H}_{\overline{R}_{n+1}(\rho)}$. $\square$

5 Terminating States and Diverging States

Definition 5 *1. For any $\rho \in \mathcal{D}(\mathcal{H})$, if $t(\rho) = \mathrm{tr}(\rho)$, then we say that ρ is a terminating state of $\mathcal{P}$.*

2. We write T for the set of terminating states of $\mathcal{P}$, that is,

$$T = \{\rho \in \mathcal{D}(\mathcal{H}) | t(\rho) = \mathrm{tr}(\rho)\}.$$

The equality $t(\rho) = \mathrm{tr}(\rho)$ is usually called the terminating condition of program $\mathcal{P}$. The intuitive meaning of this condition is that whenever the program starts in state ρ, it will terminate in a finite number of steps with probability 1. Some basic properties of terminating states are collected in the following:

Lemma 3 *1. $\rho \in T$ iff for all $R(\rho) \subseteq T$, that is, for all $f \in S_{fin}$, $\mathcal{T}_f(\rho)$ is a terminating state.*

2. Suppose that $\rho_1, \rho_2 \in \mathcal{D}(\mathcal{H})$ with $\rho_1 + \rho_2 \in \mathcal{D}(\mathcal{H})$. Then $\rho_1 + \rho_2 \in T$ iff $\rho_1 \in T$ and $\rho_2 \in T$.

3. Let $|\psi\rangle$ and $|\varphi\rangle$ be two pure states. If $|\psi\rangle, |\varphi\rangle \in T$, then any pure state $|\xi\rangle = a|\psi\rangle + b|\varphi\rangle \in T$, where $a, b \in \mathbf{C}$.

Proof:

1. The "if" part is obvious by putting $f = \epsilon$. To prove the "only if" part, we assume that $\rho \in T$. Then for any $s \in S$, it follows from Eq. (3) that

$$tr(\mathcal{T}_f(\rho)) - t_s(\mathcal{T}_f(\rho)) = tr(\rho) - \mathcal{T}_{fs}(\rho) = 0.$$

The arbitrariness of s implies that $tr(\mathcal{T}_f(\rho)) = t(\mathcal{T}_f(\rho))$.

2. If $\rho_1 \in T$ and $\rho_2 \in T$, then $t(\rho_i) = tr(\rho_i)$ $(i = 1, 2)$, and

$$\begin{aligned} tr(\rho_1 + \rho_2) \geq t(\rho_1 + \rho_2) &= \inf\{t_s(\rho_1 + \rho_2) | s \in S\} \\ &= \inf\{t_s(\rho_1) + t_s(\rho_2) | s \in S\} \geq t(\rho_1) + t(\rho_2) \\ &= tr(\rho_1) + tr(\rho_2) = tr(\rho_1 + \rho_2). \end{aligned}$$

So, $t(\rho_1 + \rho_2) = tr(\rho_1 + \rho_2)$, and $\rho_1 + \rho_2 \in T$.

Conversely, if $\rho_1 + \rho_2 \in T$, then for each $s \in S$,

$$\begin{aligned} t_s(\rho_1) + t_s(\rho_2) &= t_s(\rho_1 + \rho_2) \\ &= tr(\rho_1 + \rho_2) = tr(\rho_1) + tr(\rho_2). \end{aligned}$$

Since $t_s(\rho_i) \leq tr(\rho_i)$ $(i = 1, 2)$, it must be that $t_s(\rho_i) = tr(\rho_i)$ $(i = 1, 2)$. Therefore, $t(\rho_i) = tr(\rho_i)$, and $\rho_i \in T$ $(i = 1, 2)$.

3. Put $|\xi\rangle = a|\psi\rangle + b|\varphi\rangle$ and $|\xi'\rangle = a|\psi\rangle - b|\varphi\rangle$. Then we have:

$$\begin{aligned} &t(\xi + \xi') = t(|\xi\rangle\langle\xi| + |\xi'\rangle\langle\xi'|) \\ &= t(2|a|^2\psi + 2|b|^2\varphi) = 2|a|^2 \mathrm{tr}(\psi) + 2|b|^2 \mathrm{tr}(\varphi) \\ &= \mathrm{tr}(\xi) + \mathrm{tr}(\xi') = \mathrm{tr}(\xi + \xi'). \end{aligned}$$

Note that in the above equation, we slightly abuse the notation of density operator allowing unnormalization with trace greater than 1. This is not problematic because of linearity. So, $\xi + \xi' \in T$, and it follows from item 2 that $\xi \in T$. □

Definition 6 *1. For any $\rho \in \mathcal{D}(\mathcal{H})$, if for some schedule $s \in S$, we have $t_s(\rho) = 0$, then we say that ρ is a diverging state of $\mathcal{P}$.*

2. We write D for the set of diverging states of $\mathcal{P}$, that is,

$$D = \{\rho \in \mathcal{D}(\mathcal{H}) | t_s(\rho) = 0 \text{ for some } s \in S\}.$$

3. *We write PD for the diverging pure state of $\mathcal{P}$, that is,*

$$PD = \{|\psi\rangle \in \mathcal{H} | t_s(\psi) = 0 \text{ for some } s \in S\}.$$

The remainder of this section is devoted to examine the structure of diverging pure states PD, which is crucial in developing an algorithm for checking termination of program $\mathcal{P}$ in Sec. 7. To this end, we introduce some auxiliary notions:

Definition 7 1. *For each schedule fragment $f \in S_{fin}$, we define:*

$$PD_f = \{|\psi\rangle \in \mathcal{H} | t_f(\psi) = 0\}.$$

2. *For each $n \geq 0$, we define:*

$$PD_n = \bigcup_{f \in S_{fin}, |f| = n} PD_f.$$

3. *For each schedule $s \in S$, we define:*

$$PD_s = \{|\psi\rangle \in \mathcal{H} | t_s(\psi) = 0\}.$$

By definition, we have:

$$PD = \bigcup_{s \in S} PD_s.$$

For any $s \in S$ and $n_1 \geq n_2$, It holds that $PD_{s(\leq n_1)} \subseteq PD_{s(\leq n_2)}$ because $t_{s(\leq n_1)} \geq t_{s(\leq n_2)}$. Furthermore, we have:

$$PD_s = \bigcap_{n=0}^{\infty} PD_{s(\leq n)} \tag{4}$$

since $t_s(\cdot) = \lim_{n \to \infty} t_{s(\leq n)}(\cdot)$.

Lemma 4 *For any $f \in S_{fin}$, we have:*

1. *PD_f is a subspace of $\mathcal{H}$.*
2. *Let*

$$\mathcal{H}_0 = \{|\psi\rangle | M_0 |\psi\rangle = 0\}$$

be the orthogonal complementary subspace of measurement operator M_0. Then $PD_f \subseteq \mathcal{H}_0$.

3. *For each $k \in \{1, 2, \cdots, m\}$,*

$$PD_{kf} = H_0 \cap \mathcal{T}_k^{-1}(PD_f) \tag{5}$$

Proof:

1. Let $|\psi\rangle$, $|\varphi\rangle$ be any two states in PD_f, and $|\xi\rangle = a|\psi\rangle + b|\varphi\rangle$ be any linear superposition of them. Write $|\xi'\rangle = a|\psi\rangle - b|\varphi\rangle$. Then

$$\begin{aligned} &t_f(\xi) + t_f(\xi') = t_f(\xi + \xi') = t_f(2|a|^2\psi + 2|b|^2\varphi) \\ &= 2|a|^2 t_f(\psi) + 2|b|^2 t_f(\varphi) = 0. \end{aligned}$$

   Thus $t_f(\xi) = 0$ and $|\xi\rangle \in PD_f$.

2. Noting that $\mathcal{H}_0 = PD_\epsilon$, it is obvious.

3. For any $|\psi\rangle \in \mathcal{H}_0 \cap \mathcal{T}_k^{-1}(PD_f)$, we have $\mathrm{tr}(\mathcal{T}_k(\psi)) = \mathrm{tr}(\psi)$ for $|\psi\rangle \in \mathcal{H}_0$, and $\mathrm{supp}\mathcal{T}_k(\psi) \subseteq PD_f$ for $|\psi\rangle \in \mathcal{T}_k^{-1}(PD_f)$. Then by Eq. (1), we obtain:

$$\begin{aligned} 0 &= t_f(\mathcal{T}_k(\psi)) = \mathrm{tr}(\mathcal{T}_k(\psi)) - \mathrm{tr}(M_1\mathcal{T}_f(\mathcal{T}_k(\psi))M_1^\dagger) \\ &= \mathrm{tr}(\psi) - \mathrm{tr}(M_1\mathcal{T}_{kf}(\psi)M_1^\dagger) = t_{kf}(\psi). \end{aligned}$$

   Thus $|\psi\rangle \in PD_{kf}$. It implies that

$$\mathcal{H}_0 \cap \mathcal{T}_k^{-1}(PD_f) \subseteq PD_{kf}.$$

   Conversely, for any $|\psi\rangle \in PD_{kf} \subseteq \mathcal{H}_0$, we have $\mathrm{tr}(\mathcal{T}_k(\psi)) = \mathrm{tr}(\psi)$ and then

$$\begin{aligned} 0 &= t_{kf}(\psi) = \mathrm{tr}(\psi) - \mathrm{tr}(M_1\mathcal{T}_{kf}(\psi)M_1^\dagger) \\ &= \mathrm{tr}(\mathcal{T}_k(\psi)) - \mathrm{tr}(M_1\mathcal{T}_f(\mathcal{T}_k(\psi))M_1^\dagger) = t_f(\mathcal{T}_k(\psi)). \end{aligned}$$

   Therefore, $|\psi\rangle \in \mathcal{T}_k^{-1}(PD_f)$. We obtain:

$$PD_{kf} \subseteq \mathcal{H}_0 \cap \mathcal{T}_k^{-1}(PD_f). \ \square$$

We see that PD_s is a subspace of $\mathcal{H}_0$ for every $s \in S$ by combining Eq. (4) and Lemma 4.

Lemma 5

$$PD = \bigcap_{n=0}^{\infty} PD_n.$$

Proof: For any state $|\psi\rangle \in PD$, there is some $s \in S$ such that

$$|\psi\rangle \in PD_s \subseteq PD_{s(\leq n)} \subseteq PD_n$$

for all $n \geq 0$. Thus,

$$PD \subseteq \bigcap PD_n.$$

Conversely, we prove that $\bigcap PD_n \subseteq PD$. Suppose that $|\psi\rangle \in PD_n$ for all $n \geq 0$. Put

$$X = \{f \in S_{fin} | t_f(\psi) = 0\}.$$

Then what we need to do is to find some schedule $s \in S$ such that $s(\leq n) \in X$ for all n. To this end, put

$$E_f = \{g \in X | f \text{ is a prefix of } g\}$$

for each $f \in S_{fin}$. We consider the set

$$X' = \{f \in S_{fin} | E_f \text{ is an infinite set}\}.$$

It holds that $X' \subseteq X$ since $E_f = \emptyset$ for all $f \notin X$. So, it suffices to find some $s \in S$ such that $s(\leq n) \in X'$ for all n. Now we are going to construct such a schedule s, and our strategy is to define the head $s(\leq n)$ of s by induction on n. First, $s(\leq 0) = \epsilon \in X'$ as $E_\epsilon = X$ is an infinite set. Suppose that $s(\leq n) = s_1 s_2 \cdots s_n \in X'$ is already defined. Then there must be some $s_{n+1} \in \{1, 2, \cdots, m\}$ such that $s(\leq n+1) = s_1 s_2 \cdots s_n s_{n+1} \in X'$. This is because

$$E_{s(\leq n)} = \{s(\leq n)\} \cup \bigcup_{i=1}^{m} E_{s(\leq n)i}$$

is an infinite set, and thus at least one of $E_{s(\leq n)1}, E_{s(\leq n)2}, \cdots, E_{s(\leq n)m}$ should be an infinite set. $\square$

It is also easy to verify that for any n, $PD_{n+1} \subseteq PD_n$. On the other hand, each PD_n is the union of a finite number of subspaces of $\mathcal{H}$. The following technical lemma will help us to further clarify the structure of PD.

Lemma 6 *Suppose that X_k is the union of a finite number of subspaces of $\mathcal{H}$ for all $k \geq 1$. If $X_1 \supseteq X_2 \supseteq \cdots \supseteq X_k \supseteq \cdots$, then there exists $n \geq 1$ such that $X_k = X_n$ for all $k \geq n$.*

Proof: If for some $k \geq 1$, $X_k = \emptyset$, then the result is obvious. So we can assume that $X_k \neq \emptyset$ for all $k \geq 1$. It suffices for us to prove the lemma for a special case that X_1 is a single subspace of $\mathcal{H}$, and then the general case can be obtained by putting $X_0 = \mathcal{H}$ and considering the extended chain $X_0 \supseteq X_1 \supseteq \cdots \supseteq X_k \supseteq \cdots$.

Now, we prove the special case by induction on $\dim X_1$. First, for $\dim X_1 = 0$, $X_k = \{0\}$ for all k and the result holds. For $\dim X_1 \geq 1$, we only need to consider the nontrivial case that $X_l \neq X_1$ for some l. We choose the minimum one of such l, then $X_1 = X_2 = \cdots = X_{l-1}$ and X_l is a proper subset of X_1. Let $X_l = \bigcup_{i=1}^{b} P_i$, where $P_1, P_2, \cdots, P_b$ are subspaces of $\mathcal{H}$. Then for all $k \geq l$,

$$X_k = X_k \cap X_l = \bigcup_{i=i}^{b} (X_k \cap P_i),$$

and for each $i \in \{1, 2, \cdots, b\}$, P_i is a proper subspace of X_1 and we have $\dim P_i < \dim X_1$. Therefore, noting that $X_k \cap P_i$ is still a finite union of subspaces, by induction hypothesis, the descending chain $P_i \supseteq X_{l+1} \cap P_i \supseteq X_{l+2} \cap P_i \supseteq \cdots$

terminates at some $n_i \geq l$, that is $X_k \cap P_i = X_{n_i} \cap P_i$ for all $k \geq n_i$. Let $n = max\{n_i : i = 1, 2, \cdots, l\}$, then for all $k \geq n$ we have

$$X_k = \bigcup_{i=1}^{b}(X_k \cap P_i) = \bigcup_{i=1}^{b}(X_n \cap P_i) = X_n. \ \square$$

Now we can assert that there exists $n \geq 0$ such that $PD_k = PD_n$ for all $k \geq n$, and thus $PD = PD_n$ by combining Lemmas 4, 5 and 6. Indeed, we are able to prove an even stronger result presented in the following:

Theorem 2 *Let n be the smallest integer satisfying $PD_n = PD_{n+1}$. Then $PD = PD_n$.*

Proof: We only need to prove that for any $n \geq 0$, $PD_n = PD_{n+1}$ implies $PD_{n+1} = PD_{n+2}$. Assume that $PD_n = PD_{n+1}$ and $|\psi\rangle \in PD_{n+1}$. We are going to show that $|\psi\rangle \in PD_{n+2}$. By definition, there is $f = s_1 s_2 \cdots s_{n+1} \in S_{fin}$ such that $|\psi\rangle \in PD_f$. Put $f' = s_2 s_3 \cdots s_{n+1}$. By Eq. (5), we have $\mathrm{supp}(\mathcal{T}_{s_1}(\psi)) \subseteq PD_{f'}$. On the other hand, it follows from the assumption that

$$PD_{f'} \subseteq PD_n = PD_{n+1} = \bigcup_{g \in S_{fin}, |g| = n+1} PD_g.$$

Since $PD_{f'}$ and all PD_gs are subspaces of the finite dimensional Hilbert space $\mathcal{H}$, and a finite-dimensional Hilbert space cannot be the union of its proper subspaces (see Theorem 1.2 of [28] for reference), there must be some $g = r_1 r_2 \cdots r_{n+1} \in S_{fin}$ such that $PD_{f'} = PD_{f'} \cap PD_g$ and thus $\mathrm{supp}(\mathcal{T}_{s_1}(\psi)) \subseteq PD_g$. We have $|\psi\rangle \in \mathcal{T}_{s_1}^{-1}(PD_g)$. Furthermore, we put $g' = s_1 r_1 r_2 \cdots r_{n+1}$. Then by Eq. (5), $|\psi\rangle \in PD_{g'} \subseteq PD_{n+2}$. $\square$

6 Quantum Zero-One Law

For simplicity of presentation, from now on, we only consider normalized input state ρ, that is, we always assume that $tr(\rho) = 1$.

Definition 8 *The reachable termination probability of program $\mathcal{P}$ starting in state ρ is the infimum of termination probability of the program starting in a state reachable from ρ, that is,*

$$h(\rho) = \inf\{t(\sigma) | \sigma \in \mathcal{D}(\mathcal{H}_{R(\rho)}), \mathrm{tr}(\sigma) = 1\}.$$

The following lemma gives a characterization of terminating states in terms of reachable termination probability. It is obviously a strengthening of Lemma 3.1.

Lemma 7 *$\rho \in T$ (i.e. $t(\rho) = 1$) if and only if $h(\rho) = 1$.*

Proof: The "only if" part is obvious. To prove the "if" part, we assume that $h(\rho) = 1$. Then for any $f \in S_{fin}$, it follows from Lemma 3.1 that $t(\mathcal{T}_f(\rho)) = 1$. Since $\mathcal{T}_f(\rho)$ can be decomposed as a convex combination of its eigenvectors, by Lemma 3.2 we see that $t(\psi) = 1$ whenever $|\psi\rangle$ is an eigenvectors of $\mathcal{T}_f(\rho)$. We write:

$$V_{R(\rho)} = \{\text{eigenvectors of } \mathcal{T}_f(\rho) | f \in S_{fin}\}.$$

Then $\mathcal{H}_{R(\rho)} = \text{span}V_{R(\rho)}$, and Lemma 3.3 implies $t(\psi) = 1$ for any $|\psi\rangle \in \mathcal{H}_{R(\rho)}$. Finally, for all $\sigma \in \mathcal{D}(\mathcal{H}_{R(\rho)})$, since σ is a convex combination of pure states in $\mathcal{H}_{R(\rho)}$, we assert that $t(\sigma) = 1$ by using Lemma 3.2 once again. Therefore, $h(\rho) = 1$. □

To prove the zero-one law for reachable termination probability, we need the following technical lemma. It is obvious by definition that the reachable set is closed under $\mathcal{T}_f$, that is, $\mathcal{T}_f(R(\rho)) \subseteq R(\rho)$ for every $f \in S_{fin}$. The same conclusion is valid for the reachable space but no so obvious.

Lemma 8 *If $\rho \in \mathcal{D}(\mathcal{H}_{R(\rho)})$, then for any $f \in S_{fin}$, $\mathcal{T}_f(\rho) \in \mathcal{D}(\mathcal{H}_{R(\rho)})$.*

Proof: As $\mathcal{H}_{R(\rho)}$ is finite-dimensional, we can find a finite subset F of S_{fin} such that

$$\mathcal{H}_{R(\rho)} = \bigvee_{g \in F} supp(\mathcal{T}_g(\rho)).$$

Thus, for any $\sigma \in \mathcal{D}(\mathcal{H}_{R(\rho)})$, there exists some positive real number λ such that

$$\sigma \leq \lambda \sum_{g \in F} \mathcal{T}_g(\rho).$$

Let

$$\delta = \lambda \sum_{g \in F} \mathcal{T}_g(\rho) - \sigma \in \mathcal{D}(\mathcal{H}_{R(\rho)}).$$

Then for any $k \in \{1, 2, \cdots, m\}$,

$$\begin{aligned} \mathcal{T}_k(\sigma) + \mathcal{T}_k(\delta) = \mathcal{T}_k(\sigma + \delta) &= \mathcal{T}_k(\lambda \sum_{g \in F} \mathcal{T}_g(\rho)) \\ &= \lambda \sum_{g \in F} \mathcal{T}_{gk}(\rho) \in \mathcal{D}(\mathcal{H}_{R(\rho)}). \end{aligned}$$

So, we have $\mathcal{T}_k(\sigma) \in \mathcal{D}(\mathcal{H}_{R(\rho)})$. Moreover, we obtain $\mathcal{T}_f(\sigma) \in \mathcal{D}(\mathcal{H}_{R(\rho)})$ for any $f = k_1 \cdots k_m \in S_{fin}$ by induction on m. □

Now we are ready to present the main result in this section.

Theorem 3 (Zero-One Law) *For any ρ, we have $h(\rho) = 0$ or 1.*

Proof: We write $h = h(\rho)$ and argue that $h > 0$ implies $h = 1$. Assume $h > 0$. Then for any $\varepsilon > 0$, there exists some $\sigma \in \mathcal{D}(\mathcal{H}_{R(\rho)})$ such that $\text{tr}(\sigma) = 1$ and $h \leq t_s(\sigma) \leq h + \varepsilon$ for some $s \in S$. We can choose a sufficiently large integer

n such that $t_{s(\leq n-1)}(\sigma) \geq h/2$ because $\lim_{n\to\infty} t_{s(\leq n)}(\sigma) = t_s(\sigma) \geq h$. Applying Eq. (1), we get:

$$1 - \mathrm{tr}(T_{s(\leq n)}(\sigma)) = t_{s(\leq n-1)}(\sigma) \geq h/2. \tag{6}$$

On the other hand, we put $\lambda = tr(\mathcal{T}_{s(\leq n)}(\sigma))$. Then it follows from Lemma 8 that

$$\frac{1}{\lambda}\mathcal{T}_{s(\leq n)}(\sigma) \in \mathcal{D}(\mathcal{H}_{R(\rho)}).$$

Also, it holds that $tr[\frac{1}{\lambda}\mathcal{T}_{s(\leq n)}(\sigma)] = 1$. So, by the definition of $h(\rho)$ we have $t(\frac{1}{\lambda}\mathcal{T}_{s(\leq n)}(\sigma)) \geq h$. Consequently,

$$\begin{aligned} t_{s(>n)}(\mathcal{T}_{s(\leq n)}(\sigma)) &\geq t(\mathcal{T}_{s(\leq n)}(\sigma)) = \lambda t(\frac{1}{\lambda}\mathcal{T}_{s(\leq n)}(\sigma)) \\ &\geq \lambda h = h \cdot \mathrm{tr}(\mathcal{T}_{s(\leq n)}(\sigma)). \end{aligned}$$

Then employing Eq. (3), we obtain:

$$\begin{aligned} h + \varepsilon \geq t_s(\sigma) &= 1 - \mathrm{tr}(\mathcal{T}_{s(\leq n)}(\sigma)) + t_{s(>n)}(\mathcal{T}_{s(\leq n)}(\sigma)) \\ &\geq 1 - \mathrm{tr}(\mathcal{T}_{s(\leq n)}(\sigma)) + h \cdot \mathrm{tr}(\mathcal{T}_{s(\leq n)}(\sigma)). \end{aligned} \tag{7}$$

Now combining Eqs. (6) and (7) yields:

$$\varepsilon \geq (1-h)(1 - \mathrm{tr}(\mathcal{T}_{s(\leq n)}(\sigma))) \geq (1-h)\frac{h}{2}.$$

Finally, as ε can be arbitrarily small, it holds that $h = 1$. □

From the definition of $h(\rho)$ we see that if $h(\rho) = 1$, then termination probability $t_s(\sigma) = 1$ for any state σ in the reachable space $\mathcal{H}_{R(\rho)}$ of ρ and any schedule $s \in S$. What happens when $h(\rho) = 0$? The following proposition answers this question.

Lemma 9 *If $h(\rho) = 0$ then there exists some $\sigma \in \mathcal{D}(\mathcal{H}_{R(\rho)})$ and $s \in S$ such that $t_s(\sigma) = 0$.*

Proof: The proof is divided into three steps. First, we show that if $h(\rho) = 0$ then $t(\sigma) = 0$ for some $\sigma \in \mathcal{D}(\mathcal{H}_{R(\rho)})$. For any two states $\delta, \theta \in \mathcal{D}(\mathcal{H}_{R(\rho)})$, and any $s \in S$, we have:

$$\begin{aligned} &t(\delta) \leq t_s(\delta) = t_s(\theta) + t_s(\delta - \theta) \\ &\leq t_s(\theta) + t_s(\sqrt{(\delta-\theta)^2}) \leq t_s(\theta) + \mathrm{tr}\sqrt{(\delta-\theta)^2} \\ &= t_s(\theta) + \parallel \delta - \theta \parallel_* \overset{t_s(\theta)\to t(\theta)}{\to} t(\theta) + \parallel \delta - \theta \parallel_* \end{aligned} \tag{8}$$

Since $h(\rho) = 0$, we can find a sequence $\{\sigma_n\}$ in $\mathcal{D}(\mathcal{H}_{R(\rho)})$ such that $t(\sigma_n) \to 0$ $(n \to \infty)$. Furthermore, sequence $\{\sigma_n\}$ has an accumulation point σ because $\mathcal{D}(\mathcal{H}_{R(\rho)})$ is compact and satisfies the first countability axiom. Thus, there is a subsequence $\{\sigma_{i_k}\}$ of $\{\sigma_n\}$, which converges to σ. It follows from Eq. (8) that

$$t(\sigma) \leq t(\sigma_{i_k}) + \parallel \sigma - \sigma_{i_k} \parallel_* \to 0 \ (k \to \infty).$$

Second, we prove that if $t(\sigma) = 0$, then there exists some $1 \leq k \leq m$ such that $t(\mathcal{T}_k(\sigma)) = 0$. It suffices to see that for any $s \in S$, Eq. (3) yields:

$$\begin{aligned} 0 = t(\sigma) \geq t_s(\sigma) &= 1 - \mathrm{tr}(\mathcal{T}_{s_1}(\rho)) + t_{s(>1)}(\mathcal{T}_{s_1}(\rho)) \\ &\geq t_{s(>1)}(\mathcal{T}_{s_1}(\rho)) \geq t(\mathcal{T}_{s_1}(\rho)) \geq \min_{k=1}^{m} t(\mathcal{T}_k(\rho)) \end{aligned}$$

Third, we show that if $\sigma \in \mathcal{H}_{R(\rho)}$ satisfies $t(\sigma) = 0$ then $t_s(\sigma) = 0$ for some schedule $s \in S$. We recursively construct $s = s_1 s_2 \cdots \in S$ such that $t(\mathcal{T}_{s(\leq n)}(\sigma)) = 0$ for all $n \geq 0$. For $n = 0$, $t(\mathcal{T}_\epsilon(\sigma)) = t(\sigma) = 0$. Suppose that $s_1 s_2 \cdots s_n$ is already defined and $t(\mathcal{T}_{s_1 s_2 \cdots s_n}(\sigma)) = 0$. Then according to the conclusion in the above paragraph, we can find s_{n+1} such that $t(\mathcal{T}_{s_1 s_2 \cdots s_n s_{n+1}}(\sigma)) = t(\mathcal{T}_{s_{n+1}}(\mathcal{T}_{s_1 s_2 \cdots s_n}(\sigma))) = 0$. Finally, we get:

$$t_s(\sigma) = \sum_{n=0}^{\infty} \mathrm{tr}(M_0 \mathcal{T}_{s(\leq n)}(\sigma) M_0^\dagger) = 0$$

because $\mathrm{tr}(M_0 \mathcal{T}_{s(\leq n)}(\sigma) M_0^\dagger) \leq t(\mathcal{T}_{s(\leq n)}(\sigma)) = 0$ for all $n \geq 0$. □

7 An Algorithm for Termination Checking

A combination of the results obtained in Sec. 4, 5 and 6 leads to a necessary and sufficient condition for termination of program $\mathcal{P}$.

7.1 A Termination Condition

Theorem 4 *For any input state ρ, $\rho \in T$ (i.e. $t(\rho) = 1$) if and only if $\mathcal{H}_{R(\rho)} \cap PD = \{0\}$.*

Proof: By the zero-one law (Theorem 3) together with Lemma 7, we only need to prove that $h(\rho) = 0$ iff $\mathcal{H}_{R(\rho)} \cap PD \neq \{0\}$. If $\mathcal{H}_{R(\rho)} \cap PD \neq \{0\}$, then we arbitrarily choose $|\psi\rangle \in \mathcal{H}_{R(\rho)} \cap PD$,$\langle\psi|\psi\rangle = 1$ and it holds that $\psi \in \mathcal{D}(\mathcal{H}_{R(\rho)})$ and $t(\psi) = 0$. Thus, by definition we have $h(\rho) = 0$. Conversely, if $h(\rho) = 0$, then it follows from Lemma 9 that there exist $\sigma \in \mathcal{D}(\mathcal{H}_{R(\rho)})$ and $s \in S$ with $t_s(\sigma) = 0$. Now, let $|\psi\rangle$ be an eigenvector of σ. Then $|\psi\rangle \in \mathcal{H}_{R(\rho)}$ and $t_s(\psi) = 0$. This means that $|\psi\rangle \in \mathcal{H}_{R(\rho)} \cap PD \neq \{0\}$. □

Since we have shown in Sec. 5 that PD is a finite union of subspaces, the above condition can be checked by compute the intersections of subspaces pairs for given $\mathcal{H}_{R(\rho)}$ and PD. Therefore, an algorithm for termination checking can be obtained by simply combining an algorithms for computing reachable states and an algorithm for computing diverging pure states, which are presented in the next two subsections. An application of these algorithms to checking termination of the example program considered in Sec. 3.3 is presented in the Appendix.

7.2 An Algorithm for Computing Reachable States

Given a nondeterministic quantum program $\mathcal{P}$ and a initial state ρ, Algorithm 1 compute the reachable space $\mathcal{H}_{R(\rho)}$ based on Theorem 1.

Algorithm 1: Computing Reachable States

input : An orthonormal basis B_0 of $\mathrm{supp}(\rho)$, and a Kraus representation of $\mathcal{T}(\cdot) = \sum_{j=1}^{r} E_j \cdot E_j^\dagger$.
output: An orthonormal basis B of $\mathcal{H}_{R(\rho)}$.
set of states $B \leftarrow \emptyset$;
(* the number of elements of B *)
integer $l \leftarrow 0$;
(* the index of the state under considering *)
integer $i \leftarrow 1$;
(* put B to be B_0 initially *)
for $|x\rangle \in B_0$ **do**
  $l \leftarrow l + 1$;
  $|b_l\rangle \leftarrow |x\rangle$;
  $B \leftarrow B \cup \{|b_l\rangle\}$;
end
while $i \leq l$ **do**
  for $j \leftarrow 1$ **to** r **do**
    $|x\rangle \leftarrow E_j|b_i\rangle - \sum_{k=1}^{l} \langle b_k|E_j|b_i\rangle |b_k\rangle$;
    if $|x\rangle \neq 0$ **then**
      $l \leftarrow l + 1$;
      $|b_l\rangle \leftarrow |x\rangle / \sqrt{\langle x|x\rangle}$;
      $B \leftarrow B \cup \{|b_l\rangle\}$;
    end
  end
  $i \leftarrow i + 1$;
end
return

Correctness and complexity of the algorithm: Since B keeps to be a set of orthonormal states, $l \leq \dim \mathcal{H}$ always holds during the execution. Thus, the algorithm terminates after at most $\dim \mathcal{H}$ iterations of the **while** loop. Consider any execution of the algorithm. $B = B_0$ at the beginning, and it is convenient to write B_{i-1} for the instance of B immediately before the iteration of **while** loop for i. Then $\mathrm{span}B_i = \mathrm{span}B_{i-1} \cup \mathrm{supp}\mathcal{T}(b_i) \subseteq \mathrm{span}B_{i-1} \cup \mathcal{T}(\mathrm{span}B_{i-1})$. By Lemma 2, it is easy to prove that $\mathrm{span}B_i \subseteq \mathcal{H}_{\overline{R}_i(\rho)}$ by induction on i. Then for the output B, we have $\mathrm{span}B \subseteq \mathcal{H}_{R(\rho)}$. On the other hand, we have $\mathrm{span}B = \mathrm{span}B \cup \mathcal{T}(\mathrm{span}B)$ upon termination of the algorithm. Then $\mathcal{H}_{\overline{R}_n(\rho)} \subseteq \mathrm{span}B$ can be also proved for all n by induction. Therefore $\mathcal{H}_{R(\rho)} = \mathrm{span}B$.

To get an upper bound of the running time of the algorithm, we write $d = \dim \mathcal{H}$ and consider each iteration of the **while** loop: There are r new states

$|x\rangle$ being calculated, and each calculation is done in time $O(d^2)$ by multiplying a $d \times d$ matrix and a d-dimensional vector. Noting that $r \leq d^2$, the time complexity is $d \cdot r \cdot O(d^2) = O(d^5)$ in total. $\square$

7.3 An Algorithm for Computing Diverging Pure States

Algorithm 2 compute the set of diverging states for a given nondeterministic quantum program. The idea comes from Theorem 2: We calculate PD_n from PD_{n-1}, until the condition $PD_n = PD_{n-1}$ holds. For convenience, we write $J_n = \{PD_f : f \in S_{fin}, |f| = n\}$ and thus $\bigcup_{P \in J_n} P = PD_n$. Then to check if $PD_n = PD_{n-1}$, it suffices to check if for any $P \in J_{n-1}$, there exists $Q \in J_n$ such that $P \subseteq Q$.

Algorithm 2: Computing Pure Diverging States

input : The projection operator of $\mathcal{H}_0$.
output: A set of subspaces J_0.
(*to record J_{n-1}*)
set of subspaces $J_0 \leftarrow \emptyset$;
(*to record J_n*)
set of subspaces $J_1 \leftarrow \{\mathcal{H}_0\}$;
bool $b \leftarrow 0$;
bool $c \leftarrow 0$;
while $\neg b$ **do**
  $J_0 \leftarrow J_1$;
  $J_1 \leftarrow \emptyset$;
  for $P \in J_0$ **do**
    for $k \leftarrow 1$ **to** m **do**
      $J_1 \leftarrow J_1 \cup \{\mathcal{T}_k^{-1}(P) \cap \mathcal{H}_0\}$;
    end
  end
  (*test if $PD_n = PD_{n-1}$*)
  $b \leftarrow 1$;
  for $P \in J_0$ **do**
    $c \leftarrow 0$;
    for $Q \in J_1$ **do**
      $c \leftarrow c \vee (P \subseteq Q)$;
    end
    $b \leftarrow b \wedge c$;
  end
end
return J_0

Correctness of the algorithm: We prove by induction on n that after the nth iteration of **while** loop, J_1 becomes $\{PD_f : f \in S_{fin}, |f| = n\}$. For $n = 0$, $J_1 = \{\mathcal{H}_0\} = \{PD_\epsilon\}$. Suppose the result holds for $n - 1$. At the beginning of

the nth iteration, $J_0 \leftarrow J_1 = \{PD_f : f \in S_{fin}, |f| = n-1\}$, and then J_1 is calculated from J_0 by

$$\begin{aligned} J_1 &= \{\mathcal{H}_0 \cap \mathcal{T}_k^{-1}(P) : 1 \le k \le m, P \in J_0\} \\ &= \{\mathcal{H}_0 \cap \mathcal{T}_k^{-1}(PD_f) : 1 \le k \le m, f \in S_{fin}, |f| = n-1\} \\ &= \{PD_{kf} : 1 \le k \le m, f \in S_{fin}, |f| = n-1\} \\ &= \{PD_f : f \in S_{fin}, |f| = n\}. \end{aligned}$$

Here, $PD_{kf} = \mathcal{H}_0 \cap \mathcal{T}_k^{-1}(PD_f)$ comes from Eq. (5). So we get the correctness of the algorithm. □

It is worth noting that the termination of this algorithm comes from the descending chain condition in Lemma 6, but the terminating time n is unbounded there. So, it is still unclear how to estimate the number of iterations of the **while** loop in Algorithm 2, and consequently the complexity of computing the set of diverging states of nondeterministic quantum programs remains unsettled.

7.4 An example: Quantum Walks

This subsection is a continuation of Sec. 3.3. We show how to apply our algorithms developed above to the nondeterministic quantum program $(\{W_1, W_2\}, \{P_0, P_1\})$. In Sec. 3.3, we have shown that this program is not terminating for initial state $|0\rangle$ by observing a diverging path $1212\cdots$. Here, we give an algorithmic check for this fact.

Computing the reachable space: We use Algorithm 1 to compute the reachable space $\mathcal{H}_{R(|0\rangle\langle 0|)}$. The Kraus operators of $\mathcal{T}(\cdot)$ are $E_1 = W_1P_1$ and $E_2 = W_2P_1$. We write $B_{i,j}$ and $|x_{i,j}\rangle$ for the instance of B and $|x\rangle$ respectively, for index i and index j during the execution. Then B is calculated by a finite number of iterations of **while** loop as follows:

Initially we have

$$\begin{aligned} l &= 1, \\ |b_1\rangle &= |0\rangle, \\ B_0 &= \{|b_0\rangle\}; \end{aligned}$$

for the iteration of $i = 1$,
for $j = 1$,

$$\begin{aligned} E_1|b_1\rangle &= (|0\rangle + |1\rangle + |3\rangle)/\sqrt{3}, \\ |x_{1,1}\rangle &= (|1\rangle + |3\rangle)/\sqrt{3} \neq 0, \\ l &= 1 + 1 = 2, \\ |b_2\rangle &= (|1\rangle + |3\rangle)/\sqrt{2}, \\ B_{1,1} &= \{|b_1\rangle, |b_2\rangle\}; \end{aligned}$$

for $j = 2$,

$$\begin{aligned} E_2|b_1\rangle &= (|0\rangle - |1\rangle + |3\rangle)/\sqrt{3}, \\ |x_{1,2}\rangle &= (-|1\rangle + |3\rangle)/\sqrt{3} \neq 0, \\ l &= 2 + 1 = 3, \\ |b_3\rangle &= (-|1\rangle + |3\rangle)/\sqrt{2}, \\ B_{1,2} &= \{|b_1\rangle, |b_2\rangle, |b_3\rangle\}; \end{aligned}$$

for the iteration of $i = 2$,
for $j = 1$,

$$\begin{aligned} E_1|b_2\rangle &= (-|1\rangle + 2|2\rangle + |3\rangle)/\sqrt{6}, \\ |x_{2,1}\rangle &= 2|2\rangle/\sqrt{6} \neq 0, \\ l &= 3 + 1 = 4, \\ |b_4\rangle &= |2\rangle, \\ B_{2,1} &= \{|b_1\rangle, |b_2\rangle, |b_3\rangle, |b_4\rangle\}; \end{aligned}$$

for $j = 2$,

$$\begin{aligned} E_2|b_2\rangle &= (2|0\rangle + |1\rangle - |3\rangle)/\sqrt{6}, \\ |x_{2,2}\rangle &= 0; \end{aligned}$$

for the iteration of $i = 3$,
for $j = 1$,

$$\begin{aligned} E_1|b_3\rangle &= (-2|0\rangle + |1\rangle + |3\rangle)/\sqrt{6}, \\ |x_{3,1}\rangle &= 0; \end{aligned}$$

for $j = 2$,

$$\begin{aligned} E_2|b_3\rangle &= (-|1\rangle - 2|2\rangle - |3\rangle)/\sqrt{6}, \\ |x_{3,2}\rangle &= 0; \end{aligned}$$

for the iteration of $i = 4$,
for $j = 1$,

$$\begin{aligned} E_1|b_4\rangle &= 0, \\ |x_{4,1}\rangle &= 0; \end{aligned}$$

for $j = 2$,

$$\begin{aligned} E_2|b_4\rangle &= 0, \\ |x_{4,2}\rangle &= 0. \end{aligned}$$

So the output is

$$B = \{|0\rangle, (|1\rangle + |3\rangle)/\sqrt{2}, (-|1\rangle + |3\rangle)/\sqrt{2}, |2\rangle\},$$

and the reachable space $\mathcal{H}_{R(|0\rangle\langle 0|)} = \text{span}B$ is actually the whole state space.

Computing the set of pure diverging states: We use Algorithm 2 to compute the set of pure diverging states PD. In the algorithm, PD_f is recursively calculated by Eq. (5). Specifically, here we have

$$\mathcal{T}_k(\cdot) = W_k P_1 \cdot (W_k P_1)^\dagger \ (k = 1, 2)$$

and then the projection operator of $\mathcal{H}_0 \cap \mathcal{T}_k^{-1}(P)$ is exactly $P_1 \cap W_k^{-1} P W_k$. Now, we calculate each PD_f recursively on $|f|$ as follows:

For $|f| = 0$, we initially have

$$PD_\epsilon = P_1 = Id_4 - |2\rangle\langle 2|;$$

for $|f| = 1$,

to compute PD_1 we get that

$$W_1^{-1}|2\rangle = (|1\rangle + |2\rangle + |3\rangle)/\sqrt{3},$$
$$W_1^{-1} PD_\epsilon W_1 = \{W_1^{-1}|2\rangle\}^\perp,$$

then

$$PD_1 = P_1 \cap W_1^{-1} PD_\epsilon W_1 = |0\rangle\langle 0| + |-\rangle\langle -|,$$

where $|-\rangle = (|1\rangle - |3\rangle)/\sqrt{2}$;

to compute PD_1 we get that

$$W_2^{-1}|2\rangle = (|1\rangle + |2\rangle - |3\rangle)/\sqrt{3},$$
$$W_2^{-1} PD_\epsilon W_2 = \{W_2^{-1}|2\rangle\}^\perp,$$

then

$$PD_2 = P_1 \cap W_2^{-1} PD_\epsilon W_2 = |0\rangle\langle 0| + |+\rangle\langle +|,$$

where $|+\rangle = (|1\rangle + |3\rangle)/\sqrt{2}$;

for $|f| = 2$,

to compute PD_{11} we get that

$$W_1^{-1}|0\rangle = (|0\rangle + |1\rangle - |3\rangle)/\sqrt{3},$$
$$W_1^{-1}|-\rangle = (-|1\rangle + 2|2\rangle - |3\rangle)/\sqrt{6},$$

then

$$PD_{11} = (|0\rangle + |1\rangle - |3\rangle)(\langle 0| + \langle 1| - \langle 3|)/3;$$

to compute PD_{21} we get that

$$W_2^{-1}|0\rangle = (|0\rangle + |1\rangle + |3\rangle)/\sqrt{3},$$
$$W_2^{-1}|-\rangle = (-2|0\rangle + |1\rangle + |3\rangle)/\sqrt{6},$$

then

$$PD_{21} = P_1 \cap W_2^{-1} PD_1 W_2 = |0\rangle\langle 0| + |+\rangle\langle +|;$$

to compute PD_{12} we get that

$$W_1^{-1}|0\rangle = (|0\rangle + |1\rangle - |3\rangle)/\sqrt{3},$$
$$W_1^{-1}|+\rangle = (2|0\rangle - |1\rangle + |3\rangle)/\sqrt{6},$$

then

$$PD_{12} = P_1 \cap W_1^{-1} PD_1 W_1 = |0\rangle\langle 0| + |-\rangle\langle -|;$$

to compute PD_{22} we get that

$$W_2^{-1}|0\rangle = (|0\rangle + |1\rangle + |3\rangle)/\sqrt{3},$$
$$W_2^{-1}|+\rangle = (|1\rangle - 2|2\rangle - |3\rangle)/\sqrt{6},$$

then

$$PD_{22} = (|0\rangle + |1\rangle + |3\rangle)(\langle 0| + \langle 1| + \langle 3|)/3.$$

Since $PD_1 = PD_{12}$ and $PD_2 = PD_{21}$, we have $PD = PD_1 \cup PD_2$. Finally, we get that

$$\mathcal{H}_{R(|0\rangle\langle 0|)} \cap PD = \text{span}\{|0\rangle, |-\rangle\} \cup \text{span}\{|0\rangle, |+\rangle\} \neq \{0\}.$$

So, this program is not terminating.

8 Conclusion

In this paper, we defined a mathematic model of nondeterministic quantum programs, in which a program consists of a collection of quantum processes, each process is represented by a quantum Markov chain over the common state space, and the execution of these processes are nondeterministically scheduled. The advantage of this model is that it is independent of the details of its implementations so that we can focus our attention on examining high-level behaviors of nondeterministic quantum programs. In particular, a termination condition for nondeterministic quantum programs was found, and a classical (not quantum) algorithm for their termination checking was designed. To achieve these results, several new mathematical tools have been developed to attack the difficulty arising from the combined complexity of quantum setting and nondeterminism:

- We established a quantum zero-one law for termination probability of nondeterministic quantum programs. This law allows us to reduce the termination checking problem to emptiness checking of the intersection of the reachable space and the space of diverging pure states, instead of calculating the terminating probabilities over infinitely many execution schedules.

- We found an equivalence between the reachable space of a collection of super-operators and that of their arithmetic average.
- It was shown that the descending chain condition holds for finite unions of subspaces of a finite-dimensional Hilbert space. This helps us to extend our proof techniques for a single subspace to the case of multiple subspaces, which are unavoidable when nondeterministic choices are present.

For the further studies, an immediate topic is to extend the results presented in this paper to quantum concurrent programs where not all but only fair execution schedules are allowed. A major difficulty for such an extension comes from an essential difference between quantum concurrent programs and classical (and probabilistic) concurrent programs. In the classical case, the behavior of a concurrent program can be visualized as a directed transition graph, in which only an ordering structure determined by transition relation exists. In the state space of a quantum concurrent program, however, a linear algebraic structure and a transition relation lives together. Those methods of searching in the state space of a classical (and probabilistic) concurrent program developed in the literature (see for example [13]) are not effective in the quantum case because they usually violate the linear algebraic structure of the state space of a quantum program. It seems that a new theory of quantum graphs, where their linear algebraic and ordering structures are coordinated well, is essential for the studies of quantum concurrent programs.

Acknowledgment

We are grateful to Runyao Duan and Yuan Feng for useful discussions. This work was partly supported by the Australian Research Council (Grant No: DP110103473) and the National Natural Science Foundation of China (Grant No: 60736011).

References

[1] S. Abramsky, High-level methods for quantum computation and information, in: *Proceedings of the 19th Annual IEEE Symposium on Logic in Computer Science (LICS)*, 2004, pp. 410-414.

[2] D. Aharonov, A. Ambainis, J. Kempe and U. V. Vazirani, Quantum walks on graphs, in: *Proceedings on 33rd Annual ACM Symposium on Theory of Computing (STOC)*, 2001, pp. 50-59.

[3] T. Altenkirch and J. Grattage, A functional quantum programming language, in: *Proceedings of the 20th Annual IEEE Symposium on Logic in Computer Science (LICS)*, 2005, pp. 249-258.

[4] A. Baltag and S. Smets, LQP: the dynamic logic of quantum information, *Mathematical Structures in Computer Science*, 16(2006)491-525.

[5] S. Betteli, T. Calarco and L. Serafini, Toward an architecture for quantum programming, *European Physics Journal*, D25(2003)181-200.

[6] O. Brunet and P. Jorrand, Dynamic quantum logic for quantum programs, *International Journal of Quantum Information*, 2(2004)45-54.

[7] R. Chadha, P. Mateus and A. Sernadas, Reasoning about imperative quantum programs, *Electronic Notes in Theoretical Computer Science*, 158(2006)19-39.

[8] E. D'Hondt and P. Panangaden, Quantum weakest preconditions, *Mathematical Structures in Computer Science*, 16(2006)429-451.

[9] Y. Feng, R. Y. Duan, Z. F. Ji and M. S. Ying, M, Proof rules for purely quantum programs, *Theoretical Computer Science* 386(2007) 151-166.

[10] Y. Feng, R. Y. Duan and M. S. Ying, Bisimulation for quantum processes, in: *Proceedings of the 38th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages (POPL)*, 2011, pp. 523-534.

[11] S. J. Gay, Quantum programming languages: survey and bibliography, *Mathematical Structures in Computer Science*, 16(2006)581-600.

[12] S. J. Gay and R. Nagarajan, Communicating quantum processes, in: *Proceedings of the 32nd ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages (POPL)*, 2005, pp. 145-157.

[13] S. Hart, M. Sharir and A. Pnueli, Termination of probabilistic concurrent programs, *ACM Transactions on Programming Languages and Systems*, 5(1983)356-380.

[14] P. Jorrand and M. Lalire, Toward a quantum process algebra, in: *Proceedings of the First ACM Conference on Computing Frontiers*, 2004. pp. 111-119.

[15] G. Mitchison and R. Josza, Counterfactual computation, *Proceedings of the Royal Society of London A*, 457(2001)1175-1193.

[16] R. Nagarajan, N. Papanikolaou and D. Williams, Simulating and compiling code for the sequential quantum random access machine, *Electronic Notes in Theoretical Computer Science*, 170(2007)101-124.

[17] B. Ömer, *Structured quantum programming*, Ph.D thesis, Technical University of Vienna (2003)

[18] J. W. Sanders and P. Zuliani, Quantum programming, in: *Proceedings, Mathematics of Program Construction 2000*, LNCS 1837, pp. 80-99.

[19] P. Selinger, Towards a quantum programming language, *Mathematical Structure in Computer Science*, 14(2004)527-586.

[20] P. Selinger, A brief survey of quantum programming languages, in: *Proceedings of the 7th International SymposiumFunctional and Logic Programming (FLOPS)*, 2004, pp. 1-6.

[21] K. M. Svore, A. V. Aho, A. W. Cross, I. Chuang and I. L. Markov, A layered software architecture for quantum computing design tools, *IEEE Computer*, 39(2006)58-67.

[22] M. S. Ying, Floyd-Hoare logic for quantum programs, *ACM Transactions on Programming Languages and Systems* (accepted).

[23] M. S. Ying, R. Y. Duan, Y. Feng and Z. F. Ji, Predicate transformer semantics of quantum programs. in: S. Gay and I. Mackie (eds.), *Semantic Techniques in Quantum Computation*, Cambridge University Press, 2010, pp. 311-360.

[24] M. S. Ying and Y. Feng, Quantum loop programs, *Acta Informatica* 47(2010)221-250.

[25] M. S. Ying and Y. Feng, A flowchart language for quantum programming, *IEEE Transactions on Software Engineering*, 37(2011)466-485.

[26] P. Zuliani, Nondeterminstic quantum programming, In: P. Selinger (ed.), *Proceddings of the 2nd International Workshop on Quantum Programming Languages*. pp. 179-195.

[27] P. Zuliani, Compiling quantum programs, *Acta Informatica*, 41(2005)435-473.

[28] S. Roman, *Advanced Linear Algebra*, Springer Press, 2008.